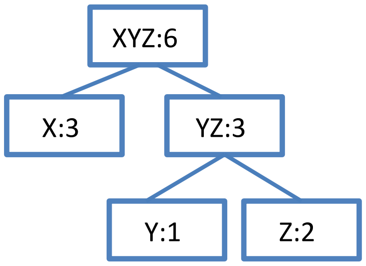
605.202 Data Structures LAB 3

Using the frequency table shown below, build a Huffman Encoding Tree. Resolve ties by giving single letter groups precedence over multiple letter groups, then alphabetically. Do not worry about punctuation or capitalization.

Print out the tree by doing a preorder traversal. Print the resulting code. An example is given below for a 3-letter alphabet. You may use any reasonable format.



X – 3 The tree in preorder is: XYZ: 6, X: 3, YZ: 3, Y: 1, Z: 2
Y – 1
Z – 2 The code is X = 0; Y = 10, Z = 11;

In your write-up, consider whether you achieved any useful data compression with this method. Compare to conventional encoding. How would your results be affected by using a different scheme to break ties, for example, if you had given precedence to alphabetical ordering and then to the number of letters in the key? What other structures did you use and why?

Encode the following strings, plus several others of your choice:

Sally sells seashells by the seashore.
Peter Piper picked a peck of pickled peppers a peck of pickled peppers Peter Piper picked.
Houston, the Eagle has landed.
Is that your final answer?

Decode the following strings:
01011001010110011111011011
1011000010101001101110110110001011001010110001011100011011 1
1111111000100011111110101111101100111111100010001111100000101000000111 0
010111

A - 19
B - 16
C - 17
D - 11

E - 42
F - 12
G – 14
H – 17
I - 16
J - 5
K - 10
L - 20
M - 19
N - 24
O - 18
P - 13
Q - 1
R - 25
S - 35
T - 25
U - 15
V - 5
W - 21
X - 2
Y - 8
Z - 3